

Aspect-Oriented Programming (AOP) and AspectJ™

AOP enables clean modular designs and code, even in the presence of cross-cutting concerns.

Typically, concerns such as performance optimizations, resource sharing, protocols etc. tangle the code of a system.

Using AspectJ, these concerns can be localized into their own clean modules.

```
class Book {
    private String title;
    private String author;
    private int pages;

    public Book(String title, String author, int pages) {
        this.title = title;
        this.author = author;
        this.pages = pages;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }

    public int getPages() {
        return pages;
    }
}
```

```
public class Printer {
    private static String status = "idle";
    private static int count = 0;

    public void print(Book book) {
        status = "printing";
        count++;
        // ... printing logic ...
        status = "idle";
    }
}

public class PrinterImpl {
    public void print(Book book) {
        Printer.print(book);
    }
}

public class BookPrinter {
    public void print(Book book) {
        Printer.print(book);
    }
}
```

Key New Ideas

- Aspects are a new class of programming construct that:
 - break the encapsulation of traditional constructs in principled ways, to
 - make it possible to cleanly capture the structure of cross-cutting concerns.

Importance and Impact

- Software development and ongoing maintenance are significant costs for DOD.
- AOP has already resulted in:
 - code size reductions (2x to >10x)
 - efficiency improvements (20% to 10x)
 - code tangling reductions (30% to 10x)
- AspectJ and AOP used by others:
 - universities, adv'd development labs
 - aspects for role modeling, business process, ...
 - several publications show other AOP systems
 - AOP workshops attract more and

Schedule
 AspectJ is Java™ extension that supports general-purpose aspect programming.

- Initial development of AspectJ, user support mechanisms, web site, tutorials etc.
- System and collateral evolution in response to feedback, building up user community